



# UNITED STATES PATENT AND TRADEMARK OFFICE



UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/092,168	03/06/2002	Sridhar Satuloori	5681-08800	9232

7590

07/19/2005

Robert C. Kowert  
Conley, Rose, & Tayon, P.C.  
P.O. Box 398  
Austin, TX 78767

EXAMINER
----------

ZHEN, LI B

ART UNIT	PAPER NUMBER
----------	--------------

2194

DATE MAILED: 07/19/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

## Office Action Summary

Application No.

10/092,168

Applicant(s)

SATULOORI ET AL.

Examiner

Li B. Zhen

Art Unit

2194

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

### Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

### Status

- 1) ☒ Responsive to communication(s) filed on 28 April 2005.
- 2a) ☒ This action is FINAL. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

### Disposition of Claims

- 4) ☒ Claim(s) 1-53 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) ☐ Claim(s) \_\_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 1-53 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

### Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on \_\_\_\_\_ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

### Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some \* c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
- \* See the attached detailed Office action for a list of the certified copies not received.

### Attachment(s)

- 1) ☐ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)  
Paper No(s)/Mail Date \_\_\_\_\_.
- 4) ☐ Interview Summary (PTO-413)  
Paper No(s)/Mail Date. \_\_\_\_\_.
- 5) ☐ Notice of Informal Patent Application (PTO-152)
- 6) ☐ Other: \_\_\_\_\_.

**DETAILED ACTION**

1. Claims 1 – 53 are pending in the current application.

***Claim Rejections - 35 USC § 103***

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

3. **Claims 1 - 53 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent Publication No. 2003/0056022 to Carlson et al. (hereinafter referred to as Carlson, cited in the previous office action) in view of "3 The Model-View-Controller Architecture," (hereinafter 3MVC, cited in the previous office action) of page 1 Figure 2: Relationship between model, view and controller objects, and the 1<sup>st</sup> – 4<sup>th</sup> paragraphs.**

4. With respect to claim 1, Carlson discloses a system, comprising:  
a processor (Carlson - col. 6:62, computer platform being any processor);  
a computer-accessible medium coupled to the processor, wherein the computer-accessible medium is configured to store program instructions executable by the processor to implement an application program (E.g. see Carlson - col. 7:8), wherein at least a first one of the application modules comprises a first dynamic component (an instance of dynamic class) and a static component (an instance of static

Art Unit: 2194

class), wherein the first dynamic component and the static component are configured to function according to an initial set of requirements (an instance of metaclass object) for the application (Carlson - col. 5:49, the attributes and methods of a Java class are classified into three categories: statically defined, dynamically defined, and configured); and

dynamic component generator configured to receive a new set of requirements for the application and generate a second dynamic component to replace the first dynamic component, wherein the second dynamic component is configured to function according to the new set of requirements (E.g. see Carlson - col. 5:14-25, the present invention allows the creation of new Java classes and the change of existing Java classes...new functionality can be introduced by configuring new classes rather than redevelopment).

Carlson does not specifically disclose the remaining features of claim 1.

However, 3MVC teaches wherein an application program comprising one or more application modules (E.g. see 3MVC, page 1, 2<sup>nd</sup> paragraph, MVC decouples them thus allowing greater flexibility and possibility for re-use).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to combine the features taught by Carlson with the 3MVC architecture to adopt the concept of decoupling an application program into multiple modules for the benefit of allowing greater flexibility and minimal interruption when requirement is altered (Carlson - col. 1:19 and col. 3:66-4:9, it would therefore be advantageous to provide a system and method for making changes in a J2EE

environment, such as added new class or altering existing classes ...with the minimal interruption... it is thus to the provision of such a system and method for defining, configuring and using dynamic, persistent Java classes).

5. With respect to claim 14, Carlson discloses a method, comprising:

installing one or more application modules comprising a static component (Carlson - col. 5:49-54, the attributes and methods of a Java class are classified into three categories: statically defined, dynamically defined, and configured; col. 6:60, a computer platform to execute the instruction; office action is taken that installation of application modules takes place prior to the execution of the instructions wherein);

one or more dynamic component generators receiving an initial set of requirements for the application modules (E.g. see Carlson - col. 5:14-25, creation of new Java classes and the change of existing Java classes through persistent configuration; office action is taken that a set of requirements is furnished prior to the creation of new Java class, which can be dynamically altered as it's being configurable); and

the one or more dynamic component generators generating (creating) one or more initial dynamic components (instances of dynamic class) for the one or more application modules, wherein the one or more initial dynamic components (instances of static class) are configured to function according to the initial set of requirements (E.g. see Carlson - col. 4:12-17, the creation and implementation of configurable Java classes on a computer platform; col. 4:22-25, the system includes a computer platform

Art Unit: 2194

executing instructions based upon... and a configurable metaclass object; col. 5:14-25, change of existing Java classes through persistent configuration; office action is taken that initial creation the configurable Java classes and the configuration of metaclass objects are to be based on the initial set of requirements).

Carlson does not specifically disclose the remaining features of claim 14.

However, 3MVC teaches wherein an application program comprising one or more application modules (E.g. see 3MVC, page 1, 1<sup>st</sup> paragraph).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to combine the features taught by Carlson with the 3MVC architecture to adopt the concept of decoupling an application program into multiple modules for the benefit of allowing greater flexibility and minimal interruption when requirement is altered (Carlson - col. 1:19; col. 3:66-4:9).

6. With respect to claim 27, Carlson discloses a method, comprising:

installing an application module, wherein at least a first one of the application modules comprises a first dynamic component and a static component, wherein the first dynamic component and the static component are configured to function according to an initial set of requirements for the application (Carlson - col. 5:49-54, the attributes and methods of a Java class are classified into three categories: statically defined, dynamically defined, and configured; col. 6:60, a computer platform to execute the instruction; office action is taken that installation of application modules takes place prior to the execution of the instructions wherein);

one or more dynamic component generators (E.g. see Carlson col. 5:14-25) receiving a new set of requirements for the application modules; and

the one or more dynamic component generators generating a new dynamic component to replace the first dynamic component, wherein the new dynamic component is configured to function according to the new set of requirements (E.g. see Carlson - col. 5:14-25, creation of new Java classes and the change of existing Java classes through persistent configuration; office action is taken that a set of requirements is furnished prior to the creation of new Java class, which can be dynamically altered as its being configurable).

Carlson does not specifically disclose the remaining features of claim 1.

However, 3MVC teaches wherein an application program comprising one or more application modules (E.g. see 3MVC, page 1, 2<sup>nd</sup> paragraph).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to combine the features taught by Carlson with the 3MVC architecture to adopt the concept of decoupling an application program into multiple modules for the benefit of allowing greater flexibility and minimal interruption when requirement is altered (Carlson - col. 1:19 and col. 3:66-4:9).

7. With respect to claim 41, Carlson discloses a tangible computer accessible medium comprising program instructions, wherein the program instructions are executable by a processor to implement an application program comprising:

wherein at least a first one of the application modules comprises a first dynamic component and a static component, wherein the first dynamic component and the static component are configured to function according to an initial set of requirements for the application (E.g. see Carlson - col. 5:49-54, the attributes and methods of a Java class are classified into three categories: statically defined, dynamically defined, and configured; Carlson -col. 5:14-25, creation of new Java classes and the change of existing Java classes through persistent configuration; office action is taken that an initial set of requirements is considered in place prior to the creation of new Java class, which can be dynamically altered as it's being configurable); and

a dynamic component generator configured to receive a new set of requirements (an altered configurations) for the application and generate a second dynamic component to replace the first dynamic component, wherein the second dynamic component is configured to function according to the new set of requirements (E.g. see Carlson - col. 4:63-5:3, allowing the reconfiguration of the configured Java class with dynamic and configured methods, and selectively altering the attributes and methods of the Java class through ...instantiated metaclass object interfaces; col. 5:14-25, change of existing Java classes through persistent configuration).

Carlson does not specifically disclose the remaining features of claim 1.

However, 3MVC teaches wherein an application program comprising one or more application modules (E.g. see 3MVC, page 1, 2<sup>nd</sup> paragraph).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to combine the features taught by Carlson with the



3MVC architecture to adopt the concept of decoupling an application program into multiple modules for the benefit of allowing greater flexibility and minimal interruption when requirement is altered (Carlson - col. 1:19 and col. 3:66-4:9).

8. With respect to claims 2, 30 and 42, the rejection of base claims 1, 27 and 41 are respectively incorporated. Carlson further discloses wherein the static/dynamic component generator does not change the static component iii response to the new set of requirements (E.g. see Carlson - col. 4:2-9, new functionality could be introduced without needing to redeploy the entire application; col. 4:63-5:3, selectively altering the attributes and methods of the java class through one or more of the instantiated metaclass object interfaces).

9. With respect to claims 3, 31 and 43, the rejection of base claims 1, 27 and 41 are respectively incorporated. Carlson further discloses wherein the dynamic component generator is configured to generate a second dynamic component to replace the first dynamic component by modifying the first dynamic component in response to the new set of requirements (E.g. see Carlson - col. 4:63-5:3, a method for altering a configurable Java class on a computer platform .... the configurable metaclass object allowing the reconfiguration of the configured java class with dynamic and configured methods, and selectively altering the attributes and methods).

10. With respect to claims 4, 32 and 44, the rejection of base claims 1, 27 and 41 are respectively incorporated. Carlson further discloses wherein the dynamic component generator is configured to replace the first dynamic component by overwriting the first dynamic component in the computer-accessible medium in response to the new set of requirements (E.g. see Carlson - col. 4:47-62, a computer program embodied on a computer-readable medium ...the program enables a configurable persistent Java class within a set of Java language instructs including a configurable metaclass object; col. 5:59-62, all existing instances will have the altered methods and attributes).

11. With respect to claims 5, 33 and 45, the rejection of base claims 1, 27 and 41 are respectively incorporated. Carlson further discloses wherein the new set of requirements is formatted according to an eXtensible Mark-up Language (XML) schema and stored in the computer-accessible medium (E.g. see Carlson - col. 4:44-46, the metaclass object can be coded in XML, Java, or any other programming language with like functionality; col. 8:15-22, an XML file can be input to the J2EE platform with the definitions of the dynamic properties and methods).

12. With respect to claims 6, 34 and 46, the rejection of base claims 1, 27 and 41 are respectively incorporated. Carlson further discloses wherein the one or more application modules comprise a second application module comprising a static component and a dynamic component (E.g. see Carlson - col. 6:58-7:6, the configurable metaclass object including a plurality of subclasses and interfaces).

13. With respect to claims 7, 35 and 47, the rejection of base claims 6, 34, 46 and 51 are respectively incorporated. Carlson further discloses wherein the dynamic component generator is configured to generate a new dynamic component for the second application module in response to receiving the new set of requirements (E.g. see Carlson - col. 3:67-4:9, making changes in a J2EE environment, such as added new class or altering existing classes, at runtime of the system with minimal interruption... new functionality could be introduced without needing to redeploy the entire application; col. 4:27, the plurality of subclasses and interface of the metaclass object include methods to selectively alter the attributes and methods of a Java class instance).

14. With respect to claims 8 and 48, the rejection of base claims 6 and 46 are respectively incorporated. Carlson further discloses wherein further comprising another dynamic component generator for the dynamic component of the second application module, wherein the other dynamic component generator is configured to generate a new dynamic component for the second application module in response to receiving a new set of requirements for the second application module (E.g. see Carlson - col. 4:63-5:3, selectively altering the attributes and methods of the Java class through one or more of the instantiated metaclass object interfaces).

15. With respect to claims 9, 36 and 49, the rejection of base claims 1, 27 and 41 are respectively incorporated. Carlson does not explicitly disclose the remaining features of claim 9.

However, 3MVC teaches wherein the first application module is a controller module, wherein the static component is a router component configured to receive user input, and wherein the dynamic component is an application logic component coupled to the router component, wherein the application logic component is configured to function according to a current set of application requirements in response to the user input (E.g. 3MVC, page 1, 4<sup>th</sup> paragraph - having data manipulated through methods that are independent of the GUI (same page 4<sup>th</sup> paragraph).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to combine the features taught by Carlson with the 3MVC architecture to further separate the application object thus allowing greater flexibility and possibility for re-use, as it is also suggested by Carlson that it would be advantageous to provide such a system and method for making changes in a J2EE environment at runtime of the system with minimal interruption. In such system and method, new functionality could be introduced without needing to redeploy the entire application, and without further manual coding and compiling of the system software. (Carlson - col. 3:66-4:9).

16. With respect to claims 10, 37 and 50, the rejection of base claims 9, 36 and 49 are respectively incorporated. Carlson further discloses wherein the application logic

component comprises an Enterprise Java Bean (EJB) session bean (E.g. see Carlson -col. 5:64-6:5, the invention accordingly consists of a design and implementation that allows adding dynamically defined as well as configured methods and attributes to an EJB preferably on the J2EE).

17. With respect to claims 11, 38 and 51, the rejection of base claims 1, 27 and 41 are respectively incorporated. Carlson does not explicitly disclose the remaining features of claim 11.

However, 3MVC teaches wherein the first application module is a model module (E.g. see 3MVC, Figure 2 module - Model; 3<sup>rd</sup> paragraph - model object knows about all the data that need to be displayed; page 1, 2<sup>nd</sup> paragraph - the goal of the MVC design pattern is to separate the application object), wherein the static component is a static data model configured to function independent of an application data representation, and wherein the dynamic component is a dynamic data model configured to function dependent upon the application data representation and according to a current set of application requirements in response to the user input (E.g. see 3MVC, page 1, 4<sup>th</sup> paragraph - data are accessed and manipulated through methods that are independent of the GUI).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to combine the features taught by Carlson with the 3MVC architecture to further separate the application object thus allowing greater flexibility and possibility for re-use, as it is also suggested by Carlson that it would be

Art Unit: 2194

advantageous to provide such a system and method for making changes in a J2EE environment at runtime of the system with minimal interruption. In such system and method, new functionality could be introduced without needing to redeploy the entire application, and without further manual coding and compiling of the system software.

(Carlson - col. 3:66-4:9).

18. With respect to claims 12, 39 and 52, the rejection of base claims 11, 24, 38 and 51 are respectively incorporated. Carlson further discloses wherein the dynamic data model comprises an Enterprise Java Bean (EJB) entity bean (E.g. see Carlson - col. 5:64-6:5, the invention accordingly consists of a design and implementation that allows adding dynamically defined as well as configured methods and attributes to an EJB preferably on the J2EE).

19. With respect to claims 13 and 40, the rejection of base claims 11 and 38 are respectively incorporated. Carlson further discloses wherein the new set of requirements indicates a change to the application data representation, and wherein the dynamic component generator is configured to generate a new dynamic data model in response to the change to the application data representation (E.g. see Carlson - col. 3:67-4:9, making changes in a J2EE environment, such as added new class or altering existing classes, at runtime of the system with minimal interruption... new functionality could be introduced without needing to redeploy the entire application; col. 4:27, the

Art Unit: 2194

plurality of subclasses and interface of the metaclass object include methods to selectively alter the attributes and methods of a Java class instance).

20. With respect to claim 15, the rejection of base claim 14 is incorporated. Carlson further discloses wherein further comprising (E.g. - see Carlson - col. 5:14-25):

receiving a new set of requirements for the application modules; and

generating one or more new dynamic components to replace the one or more initial dynamic components, wherein the one or more new dynamic components are configured to function according to the new set of requirements.

21. With respect to claim 16, the rejection of base claim 15 is incorporated. Carlson further discloses wherein said generating one or more new dynamic components comprises replacing the one or more initial dynamic components by the one or more new dynamic components by modifying the each of the one or more initial dynamic components in response to the new set of requirements (E.g. see Carlson - col. 4:63--5:3).

22. With respect to claim 17, the rejection of base claim 15 is incorporated. Carlson further discloses wherein said generating one or more new dynamic components comprises replacing the one or more initial dynamic components by the one or more new dynamic components by overwriting each of the one or more initial dynamic

Art Unit: 2194

components in a computer-accessible medium in response to the new set of requirements (E.g. see Carlson - col. 4:47-62 and col. 5:59-62).

23. With respect to claim 18, the rejection of base claim 14 is incorporated. Carlson further discloses the one or more dynamic component generators are comprised within the same application as the one or more application modules (E.g. see Carlson - col. 4:63-5:3).

24. With respect to claim 19, the rejection of base claim 14 is incorporated. Carlson further discloses wherein said generating is performed by one or more dynamic component generators comprised within an application server container, wherein the application modules are comprised within the same application server container (E.g. see Carlson - col. 1:32-34, in a business computer environment, EJB is a common server-side component architecture for the J2EE).

25. With respect to claim 20, the rejection of base claim 14 is incorporated. Carlson further discloses wherein said generating, the static components comprised by the one or more application modules are not changed in response to the new set of requirements (E.g. see Carlson - col. 4:2-9 and col. 4:63-5:3).

26. With respect to claim 21, the rejection of base claim 14 is incorporated. Carlson further discloses wherein the new set of requirements is formatted according to an



Art Unit: 2194

eXtensible Mark-up Language (XML) schema (E.g. see Carlson - col. 4:44-46 and col. 8:15-22).

27. With respect to claim 22, the rejection of base claim 14 is incorporated. Carlson does not explicitly disclose the remaining features of claim 9. However, 3MVC teaches wherein one of the one or more application modules is a controller module, wherein the static component is a router component configured to receive user input, and wherein a dynamic component generated for the one of the one or more application modules is an application logic component coupled to the router component, wherein the application logic component is configured to function according to a current set of requirements in response to the user input (E.g. see 3MVC, page 1, 4<sup>th</sup> paragraph).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to combine the features taught by Carlson with the 3MVC architecture to further separate the application object thus allowing greater flexibility and possibility for re-use, as it is also suggested by Carlson that it would be advantageous to provide such a system and method for making changes in a J2EE environment at runtime of the system with minimal interruption. In such system and method, new functionality could be introduced without needing to redeploy the entire application, and without further manual coding and compiling of the system software. (Carlson - col. 3:66-4:9).

28. With respect to claim 23, the rejection of base claim 22 is incorporated. Carlson further discloses wherein the application logic component comprises an Enterprise Java Bean (EJB) session bean (E.g. see Carlson - col. 5:64-6:5).

29. With respect to claim 24, the rejection of base claim 14 is incorporated. Carlson does not explicitly disclose the remaining features of claim 24.

However, 3MVC teaches wherein one of the one or more application modules is a model module (E.g. see 3MVC, Figure 2 module - Model; 3<sup>rd</sup> paragraph, and page 1, 2<sup>nd</sup> paragraph), wherein the static component is a static data model configured to function independent of an application data representation, and wherein a dynamic component generated for the one of the one or more application modules is a dynamic data model configured to function dependent upon the application data representation and according to a current set of requirements in response to the user input (E.g. see 3MVC, page 1, 4<sup>th</sup> paragraph).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to combine the features taught by Carlson with the 3MVC architecture to further separate the application object thus allowing greater flexibility and possibility for re-use, as it is also suggested by Carlson that it would be advantageous to provide such a system and method for making changes in a J2EE environment at runtime of the system with minimal interruption. In such system and method, new functionality could be introduced without needing to redeploy the entire

application, and without further manual coding and compiling of the system software.  
(Carlson - col. 3:66-4:9).

30. With respect to claim 25, this is similar in scope to claims 12, 39 and 52; therefore, this is rejected for the similar reasons as claims 12, 39 and 52 above.

31. With respect to claim 26, the rejection of base claim 24 is incorporated. Carlson further discloses wherein further comprising (E.g. see Carlson - col. 3:67-4:9 and Col. 4:27):

receiving a new set of requirements indicating a change to the application data representation; and

generating a new dynamic data model in response to the change to the application data representation.

32. With respect to claim 28, the rejection of base claim 27 is incorporated. Carlson further discloses wherein the one or more dynamic component generators are comprised within the same application server as the one or more application modules (E.g. see Carlson - col. 1:32-34).

33. With respect to claim 29, the rejection of base claim 27 is incorporated. Carlson further discloses wherein the one or more application modules are comprised within the same application server container (E.g. see Carlson - col. 1:32-34).

34. With respect to claim 53, this is similar in scope to claims 7, 35 and 47 above; therefore, this is rejected for the similar reasons as claims 7, 35 and 47 above.

***Response to Arguments***

35. Applicant's arguments filed April 28, 2005 have been fully considered but they are not persuasive.

In response to the Non-Final office action dated January 26, 2005, applicant argues:

(1) Carlson in view of MVC fails to teach or suggest a dynamic component generator configured to receive a new set of requirements (p. 14, lines 16 – 18);

(2) providing a system that includes methods to alter attributes and methods of Java class instance is not the same as application having a generator component that receives a set of requirements and Carlson does not mention any component of this system receiving a set of requirements (p. 15, lines 5 – 8); and

(3) Carlson in view of MVC also fails to teach or suggest a dynamite component generator configured to generate a second dynamic component to replace the first dynamic component (p. 15, lines 12 – 15).

As to argument (1), examiner respectfully disagrees and notes that Carlson teaches a dynamic component generator (E.g. see Carlson – p. 3, paragraph 0037, Softype Interface 20 supports the DynamicEntityBean 24, and contains the methods to allow run-time configuration of the Softype instances the attributes and methods for a

class are stored in the "Softtype" meta-class object such that each SoftType instance represents a particular configuration and acts as a factory for the instantiation of concrete instances) configured to receive a new set of requirements (E.g. see Carlson – p. 4, paragraph 0041, an XML file can be input to the J2EE platform with the definitions of the dynamic properties and methods; see also rejection to claims 5, 33 and 45 above). Examiner notes that the SoftType is a meta-class object that dynamically generates components (E.g. see Carlson – p. 3, paragraph 0037, each SoftType instance represents a particular configuration and acts as a factory for the instantiation of concrete instances); therefore, the SoftType meta-class object corresponds to the dynamic component generator. In addition, examiner notes that definitions of the dynamic properties and methods correspond to the set of requirements (E.g. see Carlson – p. 4, paragraph 0041 and the rejection to claims 5, 33 and 45 above).

In response to argument (2), examiner respectfully disagrees and notes that the SoftType object includes a DynamicEntity that includes the methods to add and remove directly contained properties, and also add and remove methods from a class object that instantiates Softtype (E.g. see Carlson – p. 3, paragraph 0036) and Carlson notes that one of skill in the art would be familiar with several tools for XML input into a J2EE platform (E.g. see Carlson - p. 4, paragraph 0041).

As to argument (3), examiner respectfully disagrees and notes that the SoftType meta-class object corresponds to the dynamic component generator (E.g. see Carlson – p. 3, paragraph 0037 and response to argument (1) above).

***Conclusion***

36. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire **THREE MONTHS** from the mailing date of this action. In the event a first reply is filed within **TWO MONTHS** of the mailing date of this final action and the advisory action is not mailed until after the end of the **THREE-MONTH** shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than **SIX MONTHS** from the mailing date of this final action.


37. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Li B. Zhen whose telephone number is (571) 272-3768. The examiner can normally be reached on Mon - Fri, 8:30am - 5pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on (571) 272-3756. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Li B. Zhen  
Examiner  
Art Unit 2194

lbz

  
MENG-AL T. AN  
SUPERVISORY PATENT EXAMINER  
TECHNOLOGY CENTER 2194